IN THE CLAIMS:

Please cancel claims amend the claims as follows:

1. (Previously Presented) A method for determining the possibility of adverse effect arising from a code change in a computer program, comprising the steps of:

identifying important classes within a computer program;

determining directly and indirectly dependent classes of said important classes, wherein said important classes comprise superclasses of said directly and indirectly dependent classes;

associating test cases with said important classes and with said directly and indirectly dependent classes; and

for a given code change to first important class:

running all test cases associated with said first important class and associated with dependent classes of said first important class; and

indicating the possibility of an adverse effect if any run test case fails.

2. (Currently Amended) The method according to claim 1, ~~all the limitations of which are incorporated herein by reference,~~ wherein the identification of important classes includes building an inheritance structure of class names and super classes of said program, and from which said structure start points and direct and indirect descendants thereof are identified.

3. (Currently Amended) The method according to claim 2, ~~all the limitations of which are incorporated herein by reference,~~ wherein determining dependent classes includes:

finding references in said program to said important classes;

finding methods invoked by said important classes; and

determining a dependency structure of said methods that incorporates said dependent classes.

4. (Currently Amended) The method according to claim 3, ~~all the limitations of which are incorporated herein by reference,~~ wherein determining said dependency structure further includes identifying both directly dependent and indirectly dependent classes, said indirectly dependent classes exhibiting a producer/consumer relation for persistent data.

5. (Currently Amended) The method according to claim 1, ~~all the limitations of which are incorporated herein by reference,~~ wherein indicating an adverse step includes generating a program output.

6. (Previously Presented) A method for determining the possibility of adverse effect arising from a code change in a computer program having a plurality of classes, comprising the steps of:

identifying important ones of said classes;

determining directly and indirectly dependent classes of said important classes, wherein said important classes comprise superclasses of said directly and indirectly dependent classes;

associating test cases with said classes and with said directly and indirectly dependent classes; and

for a given code change to a first class:

running all test cases associated with said first class and associated with dependent classes of said first class; and

indicating the possibility of an adverse effect if any run test case fails.

7. (Currently Amended) The method according to claim 6, ~~all the limitations of which are incorporated herein by reference,~~ wherein the identification of important classes includes building an inheritance structure of class names and super classes of said program, and from which said structure start points and direct and indirect descendants thereof are identified.

8. (Currently Amended) The method according to claim 7, ~~all the limitations of which are incorporated herein by reference,~~ wherein determining dependent classes includes:

finding references in said program to said important classes;

finding methods invoked by said important classes; and

determining a dependency structure of said methods that incorporates said

dependent classes.

9. (Currently Amended) The method according to claim 8, ~~all the limitations of which are incorporated herein by reference,~~ wherein determining said dependency structure further includes identifying both directly dependent and indirectly dependent classes, said indirectly dependent classes exhibiting a producer/consumer relation for persistent data.

10. (Currently Amended) The method according to claim 6, ~~all the limitations of which are incorporated herein by reference,~~ wherein indicating an adverse step includes generating a program output.

11. (Previously Presented) A data processing system comprising:

a memory storing a program, the program having a plurality of classes;

a user input by which program code changes can be made;

a processor operable to

identify important ones of said classes, determine directly and indirectly dependent classes of said important classes, wherein said important classes comprise superclasses of said directly and indirectly dependent classes, and

associate test cases with said classes and with said directly and indirectly dependent classes, and

wherein for a given code change to a first class input via said user input, said

processor runs all test cases associated with said first class and associated with dependent classes of said first class; and

further comprising:

an output means by which a program output is generated to indicate the possibility of an adverse effect if any run test case fails.

12. (Currently Amended) The data processing system according to claim 11, ~~all the limitations of which are incorporated herein by reference,~~ wherein said processor is further operable to identify important classes by building an inheritance structure of class names and super classes of said program, and from which said structure start points and direct and indirect descendants thereof are identified.

13. (Currently Amended) The data processing system according to claim 12, ~~all the limitations of which are incorporated herein by reference,~~ wherein said processor is further operable to determine dependent classes by finding references in said program to said important classes, finding methods invoked by said important classes, and determining a dependency structure of said methods that incorporates said dependent classes.

14. (Currently Amended) The data processing system according to claim 13, ~~all the limitations of which are incorporated herein by reference,~~ wherein said processor is further operable to determine said dependency structure by identifying both directly

dependent and indirectly dependent classes, said indirectly dependent classes exhibiting a producer/consumer relation for persistent data.

15. (Previously Presented) A computer program product comprising a computer program carried on a storage medium, said computer program comprising:

a program code element having a plurality of classes;

a code element for identifying important ones of said classes;

a code element for determining directly and indirectly dependent classes of said important classes, wherein said important classes comprise superclasses of said directly and indirectly dependent classes;

a code element for associating test cases with said classes and with said directly and indirectly dependent classes;

a code element for running all test cases associated with a first class and associated with dependent classes of said first class contingent upon a given code change to said first class within said program code element; and

a code element for indicating the possibility of an adverse effect if any run test case fails.

16. (Previously Presented) A computer program product comprising computer program carried on a storage medium, said computer program comprising:

a program code element having a plurality of classes;

a code element for identifying important classes within a computer program;

a code element for determining directly and indirectly dependent classes of said important classes, wherein said important classes comprise superclasses of said directly and indirectly dependent classes;

a code element for associating test cases with said important classes and with said directly and indirectly dependent classes;

a code element for running all test cases associated with a first class and associated with dependent classes of said first class contingent upon a given code change to said first class within said program code element; and

a code element for indicating the possibility of an adverse effect if any run test case fails.

17. (Currently Amended) The method according to claim 1, ~~all the limitations of which are incorporated herein by reference,~~ wherein said directly and indirectly dependent classes of a given important class directly refer to said given important class or consume data from said given important class.

18. (Currently Amended) The ~~method~~ computer program product according to claim 16, ~~all the limitations of which are incorporated herein by reference,~~ wherein said directly and indirectly dependent classes of a given important class directly refer to said given important class or consume data from said given important class.

19. (Currently Amended) The data processing system according to claim 11, ~~all the limitations of which are incorporated herein by reference,~~ wherein said directly and indirectly dependent classes of a given important class directly refer to said given important class or consume data from said given important class.

20. (Currently Amended) The computer program product according to claim 15, ~~all the limitations of which are incorporated herein by reference,~~ wherein said directly and indirectly dependent classes of a given important class directly refer to said given important class or consume data from said given important class.